eXo Platform 3.0 - CMIS Developer Guide

Chapter 1. Introduction

The eXo Platform provides CMIS support using the xCMIS project and the WCM Storage provider.

1.1. About CMIS

The CMIS standard aims at defining a common content management web services interface that can be applied in content repositories and bring about the interoperability across repositories. The formal specification of CMIS standard is approved by the Organization for the Advancement of Structured Information Standards (OASIS) technical committee who drives the development, convergence and adoption of global information society. With CMIS, enterprises now can deploy systems independently, and create specialized applications running over a variety of content management systems.

To see the advantages of content interoperability and the significance of CMIS as a whole, it is necessary to learn about mutual targets which caused the appearance of specification first.

1. Content integration: With CMIS, integrating content among various repositories, even those created by different vendors in a single application, becomes faster, simpler and more effective. CMIS makes it possible for customers to integrate content management systems into their key business processes across business departments and vendor implementations.
2. Access unification: CMIS enables different applications and manufacturers to be connected to a CMIS-enabled content repository simply. With CMIS, a business application's developer can focus on the application's business logic, rather than issues related to the compatibility or content migration.

1.2. About xCMIS

xCMIS project, which is initially contributed to the Open Source community by eXo Platform, is an Open Source implementation of the Content Management Interoperability Services (CMIS) specification. xCMIS supports all the features stated in the CMIS core definition and both REST AtomPub and Web Services (SOAP/WSDL) protocol bindings.

Tip

To learn more about xCMIS, visit:

- [http://gazarenkov.blogspot.com/2010/01/xcmis1-cmis-in-nutshell.html]
- [http://code.google.com/p/xcmis/]

1.3. About eXo CMIS

eXo CMIS is built on the top of xCMIS embedded in Platform to expose the WCM drives as the CMIS repositories. The CMIS features are implemented as a set of components deployed on the eXo Container using XML files to describe the service configuration.

Note

SOAP protocol binding is not implemented in eXo CMIS.

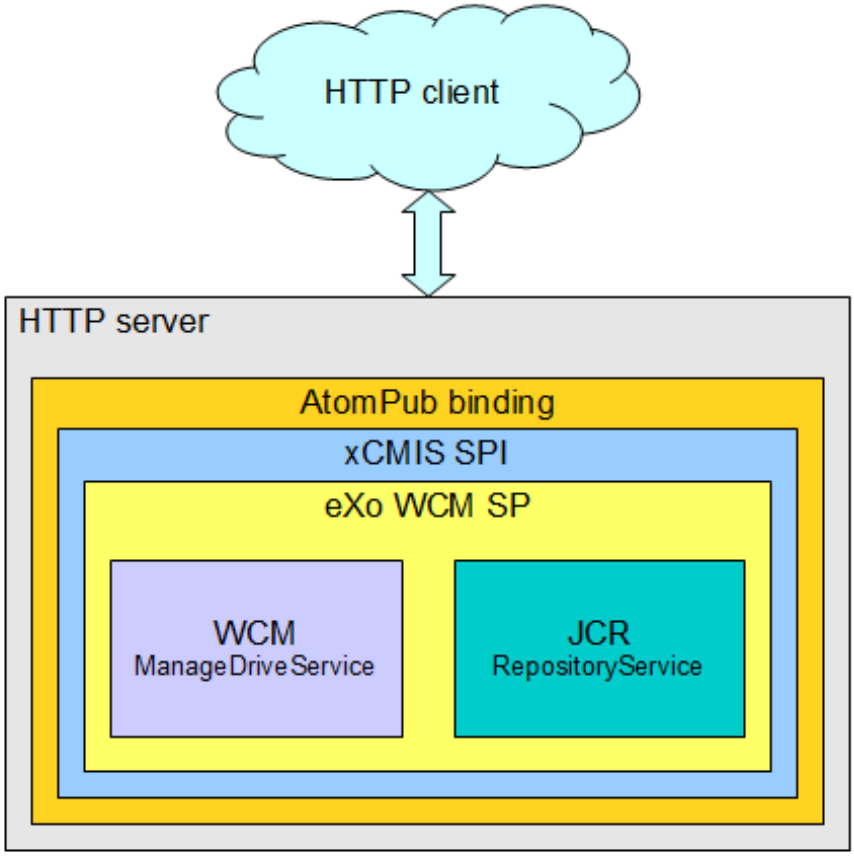

Figure: how eXo CMIS works

WCM drives exposure is implemented as WCM storage provider to xCMIS SPI. Storage provider uses mappings from WCM's *ManageDriveService* to actual JCR nodes. And *AtomPub* bindings makes WCM structure available via CMIS standard API.

Chapter 2. CMIS specification

Note

This is related to Content Management Interoperability Services (CMIS) Version 1.0 OASIS Standard 1 May 2010

[http://en.wikipedia.org/wiki/Content_Management_Interoperability_Services]

The CMIS interface is designed to be layered on top of existing Content Management systems and their existing programmatic interfaces. It is intended to expose all of the CM systems capabilities through the CMIS interfaces exhaustively. The CMIS specification defines the followings:

- A standard "domain model" for an ECM system - a set of core concepts included in all modern ECM systems, such as Object Types, properties, folders, documents, versions, and relationships; and a set of operations performed on those concepts, such as updating documents, or navigating via a folder hierarchy.
- The way to bind the CMIS domain model to two different web service protocols, including the Simple Object Access Protocol (SOAP) used in many ECM systems, and the Atom used in many Web 2.0 applications.

Note

SOAP protocol is not implemented in eXo CMIS.

CMIS specification provides a Web services interface which can:

- Work over existing repositories, enabling customers to build and leverage applications against multiple repositories.
- Decouple Web services and content from the content management repository, enabling customers to manage content independently.
- Provide common Web services and Web 2.0 interfaces to dramatically simplify the application development.
- Build the development platform and language agnostic.
- Support the composite application development and mashups by the business or IT analysts.

Chapter 3. xCMIS project

xCMIS includes the client side frameworks for integrating content from different enterprise repositories, according to http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=cmis .

The project is to make joining Enterprise Content repositories simpler by offering CMIS ablity and exposing them to language-independent CMIS clients via the most convenient protocol.

xCMIS project:

- Is embedded, packaged as the J2EE Web archive (WAR) and prepared "download and go" Tomcat bundle.
- Has a live demo with the full-featured CMIS Expert client, which is accessible on the xcmis.org site and with prepared "download and go" Tomcat bundle (the client is accessible as the remote gadget).
- Is embedded in Platform to create the special xCMIS jcr repository and access it with any CMIS client.
- Tested with third-party CMIS clients, such as IBM CMIS Firefox Connector and CMIS Spaces Flex+AIR client. Either local repository (as described here) or http://xcmis.org/xcmis1/rest/cmisatom can be used as a CMIS repository's endpoint URL for these or other types of clients.

Benefits of xCMIS

- xCMIS is an open source, server side Java CMIS implementation, enabling to expose content in the existing content repositories according to the protocols defined in the CMIS specification.
- xCMIS will give developers a way to make their content repositories "pluggable" on the server side based on the internal Storage Provider Interface and additional protocol on-demand bindings.
- xCMIS will provide (several) CMIS client frameworks for repository-application and repository-repository interactions. The programming language and supported protocol can be selected by users. For example, the reasonable choice for using web applications, gadgets, and/or mashups is JavaScript or GWT over REST AtomPub, while for inter-repository exchange, it may be Java over Web Services, i.e. WSDL/SOAP.)
- Both the server and client sides of xCMIS are easily integrated in the eXo Platform 3.0 infrastructure. In particular, xCMIS exposes the eXo JCR content repository and provides a framework for building web applications and gadgets for the GateIn portal.

The xCMIS project is distributed under the LGPL license. You can download sources on http://code.google.com/p/xcmis/source/checkout , or visit http://code.google.com/p/xcmis/w/list for more information.

Chapter 4. CMIS features

The CMIS specification prescribes:

- Domain model
- Services

4.1. CMIS Domain Model

The CMIS Domain Model defines a repository as a container and an entry point to the objects that is quite simple and non-restrictive. The followings are some of the common entities of the domain model.

- Repository is a container of objects with a set of "capabilities" which may be different depending on implementation.
- Object is the entity managed by a CMIS repository.
- Object Type is a classification related to an object. It specifies a fixed and non-hierarchical set of properties ("schema") that all objects of that type have.
- Document Object is an elementary information entity.
- Folder Object is a collection of fileable objects.
- Relationship Object is used to describe a dependent object semantically.
- Policy Object represents an administrative policy applied to an object.
- Access Object defines permissions.
- Versioning is to support versioning for Document objects.
- Query is type-based in a simplified SQL SELECT statement.
- Change Log is a mechanism which enables applications to discover changes to the objects stored.

Note

CMIS specifies a query language based on the SQL-92 standard, plus the extensions, in conjunction with the model mapping defined by the CMIS's relational view.

All objects are classified by an Object Type which declares that all objects of the given type have some sets of properties in common. Each property consists of a set of attributes, such as the TypeID, the property ID, its display name, its query name, and more). There are only four base types, including Document, Folder, Relationship, and Policy. Also, you can extend those basic types of modifying a set of their properties.

Document Object and Folder Object are self-explanatory. Document Object has properties to hold document metadata, such as the document author, modification date and custom properties. It can also contain a content stream whether it is required, and renditions, such as a thumbnail view of document. Folder is used to contain objects. Apart from the default hierarchical structure, CMIS can optionally store objects in multiple folders or

in no folders at all (so-called multifiling and unfiling capabilities). Relationship Object denotes the connection between two objects (target and source). An object can have multiple relationships with other objects. Policy Object is a way to define administrative policies in managing objects. For example, you can use a CMIS policy to define which documents are subject to retention policies.

4.2. CMIS Services

CMIS provides a set of services to access and manage the content or repository. These services include:

- Repository Services: To discover information about the repository and the object types defined for the repository.
- Navigation Services: To traverse the folder hierarchy in a CMIS repository, and to locate documents which are checked out.
- Object Services: To execute ID-based CRUD functions (Create, Retrieve, Update, Delete) on objects in a repository.
- Multi-filing Services (optional): To put an object in more than one folder (multi-filing) or outside the folder hierarchy (unfiling).
- Discovery Services: To search for queryable objects in a repository.
- Versioning Services: To check out, navigate to documents or update a Document Version Series (checkOut, cancelCheckOut, getPropertiesOfLatestVersion, getAllVersions, deleteAllVersions).
- Relationship Services (optional): To retrieve an object for its relationships.
- Policy Services (optional): To apply, remove, or query for policies.
- ACL Services: To return and manage the Access Control List (ACL) of an object. ACL Services are not supported by all repositories.

Some repositories might not implement certain optional capabilities, but they are still considered CMIS-compliant. Each service has binding which defines the way messages will be serialized and wired. Binding are based on HTTP and uses the Atom Publishing Protocol.

4.3. Integration with eXo WCM

eXo Web Content Management (WCM) system provides CMIS access to its content storage features:

- WCM drives
- Document files and folders
- Symlinks
- Categories

To expose WCM drives as CMIS repositories there is special extension of *CmisRegistry*. Read admin guide for configuration of *org.exoplatform.ecms.xcmis.sp.jcr.exo.DriveCmisRegistry* component.

Work with CMIS is based on reference documents returned by services. Each CMIS service returns response containing links to other services describing the Document or operations on it. In most cases a Document will be asked by its ID. Some services accepts a Document path.

Note

Notes for use cases: To access the eXo CMIS services from the client side, use the Curl tool [http://curl.haxx.se/download.html]. CMIS AtomPub binding which is based upon the Atom (RFC4287) and Atom Publishing Protocol (RFC5023) will be used.

SOAP binding is not implemented in the eXo Platform 3.x.

4.3.1. JCR namespaces and nodetypes

eXo CMIS uses special JCR namespaces *cmis* and *xcmis* internally.

To expose drives content following nodetypes supported:

- nt:file nodetype for representation of cmis:documents
- nt:folder for representation of cmis:folder

Since CMIS specification not allows to have more root types except of described above (cmis:documents and cmis:folder), those two (nt:file and nt:folder) are mapped to CMIS types.

There is two more node types which are used: cmis:policy and cmis:relationship which represent CMIS types with corresponded (see Services description for details).

Additionally nodetypes used in WCM are mapped as follow:

- nt:unstructured + extentions as cmis:folder
- exo:taxonomy + extentions as cmis:folder

In other words only nodetypes extending nt:file, nt:folder, nt:unstructured and exo:taxonomy will be exposed correctly via CMIS API.

Warning

WCM's nodetype exo:article isn't supported by eXo CMIS due to not compliant structure to nt:file.

4.3.2. WCM drives as CMIS Repositories

The WCM drive is used to expose as an isolated repository via the CMIS service. Operations on the repository will reflect the drive immediately.

Tip

Working with CMIS repositories it's important understand that a repository reflects a WCM Drive, which is a sub-tree in JCR workspace. And two or more drives can be mapped to a same workspace or a sub-tree. As a result changes in one repository can affect others. Consult with WCM drives

mappings to know actual location of a content you'll access or change.

4.3.2.1. Use Case: Browse Drives via *getRepository*

- Login to the website as a user with the developer role.
- Open **Group | Sites Explorer**, you can see the drives set of WCM, such as **Sites Management**, **DMS Administration**.
- Get the list of these WCM drives via CMIS using *Curl*, asking *getRepositories* service:

```
curl -o getrepos.xml -u root:gtn http://localhost:8080/rest/private/cmisatom/
```

The requested file (getrepos.xml) contains the set of Repositories in the AtomPub format. The root element represents the set of **workspaces** representing **WCM drives** related to resources, for example for **DMS Administration** the response will contains data like:

```
<service xmlns="http://www.w3.org/2007/app" xmlns:atom="http://www.w3.org/2005/Atom" xmlns:cmisra="http://doc
<workspace>
   <atom:title type="text">DMS Administration</atom:title>
   <cmisra:repositoryInfo xmlns:cmis="http://docs.oasis-open.org/ns/cmis/core/200908/">
      <cmis:repositoryId>DMS Administration</cmis:repositoryId>
      <cmis:repositoryName>DMS Administration</cmis:repositoryName>
   </cmisra:repositoryInfo>
   <collection href="http://localhost:8080/rest/private/cmisatom/DMS%20Administration/query">
      <atom:title type="text">Query</atom:title>
      <cmisra:collectionType>query</cmisra:collectionType>
   </collection>
   <collection href="http://localhost:8080/rest/private/cmisatom/DMS%20Administration/children/00exo0jcr0root
      <atom:title type="text">Folder Children</atom:title>
      <cmisra:collectionType>root</cmisra:collectionType>
   </collection>
   <collection href="http://localhost:8080/rest/private/cmisatom/DMS%20Administration/checkedout">
      <atom:title type="text">Checkedout collection</atom:title>
      <cmisra:collectionType>checkedout</cmisra:collectionType>
   </collection>
   <collection href="http://localhost:8080/rest/private/cmisatom/DMS%20Administration/unfiled">
      <atom:title type="text">Unfiled collection</atom:title>
      <cmisra:collectionType>unfiled</cmisra:collectionType>
   </collection>
   <collection href="http://localhost:8080/rest/private/cmisatom/DMS%20Administration/types">
      <atom:title type="text">Types Children</atom:title>
      <cmisra:collectionType>types</cmisra:collectionType>
   </collection>
   <cmisra:uritemplate>
      <cmisra:template>http://localhost:8080/rest/private/cmisatom/DMS%20Administration/object/{id}?filter={f
      </cmisra:template>
      <cmisra:type>objectbyid</cmisra:type>
      <cmisra:mediatype>application/atom+xml;type=entry</cmisra:mediatype>
   </cmisra:uritemplate>
   <cmisra:uritemplate>
      <cmisra:template>http://localhost:8080/rest/private/cmisatom/DMS%20Administration/objectbypath?path={pa
      </cmisra:template>
      <cmisra:type>objectbypath</cmisra:type>
      <cmisra:mediatype>application/atom+xml;type=entry</cmisra:mediatype>
   </cmisra:uritemplate>
   <cmisra:uritemplate>
      <cmisra:template>http://localhost:8080/rest/private/cmisatom/DMS%20Administration/query?q={q}&amp;searc
      </cmisra:template>
      <cmisra:type>query</cmisra:type>
      <cmisra:mediatype>application/atom+xml;type=feed</cmisra:mediatype>
   </cmisra:uritemplate>
   <cmisra:uritemplate>
      <cmisra:template>http://localhost:8080/rest/private/cmisatom/DMS%20Administration/typebyid/{id}
      </cmisra:template>
```

```
        <cmisra:type>typebyid</cmisra:type>
        <cmisra:mediatype>application/atom+xml;type=entry</cmisra:mediatype>
      </cmisra:uritemplate>
    </workspace>
  </service>
```

Here are the collection of services and predefined templates which can be used from the client side to request resources related to this repository. For example, to get the WCM node of **DMS Administration** drive by path, the **objectbypath** template can be used:

```
http://localhost:8080/rest/private/cmisatom/DMS%20Administration/objectbypath?path={path}&amp;filter={filter}&am
```

where parameters include:

- Required:
  - ID repositoryId: The identifier for the repository.
  - String path: The path to the object.
- Optional:
  - String filter
  - Boolean includeAllowableActions
  - Enum includeRelationships
  - String renditionFilter
  - Boolean includePolicyIds
  - Boolean includeACL

Note

Find full description of all sepcefied services in the CMIS specefication.

4.3.3. WCM Symlinks

Symlinks are used to organize the virtual access to documents in WCM, which is implemented like links in Unix/Linux/Mac OS (refer to ln command for more details).

Note

JCR exo:symlink nodetype used for such type nodes.

4.3.3.1. Use Case: Follow Symlinks

- Login to the acme website as a user with the developer role.
- Open **Group | Sites Explorer**, select **Sites Management**, go to folder /acme/documents.

- Upload any file (e.g. test.txt) /acme/documents
- Add this file to **acme/News** category. It will create symlink to /acme/documents/test.txt in /acme/categories/acme/News
- Get content of folder /acme/categories/acme/News via CMIS:

```
curl -o news.xml -u root:gtn http://localhost:8080/rest/private/cmisatom/Managed%20Sites/objectbypath?path=/acme
```

The requested file (products.xml) contains the entry with information about the folder.

- The list of properties for the object (such as node Id or type).
- Allowable Actions can be performed on the document; for example, requesting the children list.
- ACL and policies information.

```
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:cmisra="http://docs.oasis-open.org/ns/cmis/restatom/200908/">
   <id>f59a3539c0a80003625790bdadf566c5</id>
   <published>2010-09-09T18:11:57.707Z</published>
   <updated>2010-09-09T18:11:57.707Z</updated>
   <summary type="text"/>
   <author>
      <name>system</name>
   </author>
   <title type="text">News</title>
   <link href="http://localhost:8080/rest/private/cmisatom/Managed%20Sites" rel="service" type="application/atom
   <link href="http://localhost:8080/rest/private/cmisatom/Managed%20Sites/object/f59a3539c0a80003625790bdadf566
   <link href="http://localhost:8080/rest/private/cmisatom/Managed%20Sites/object/f59a3539c0a80003625790bdadf566
   <link href="http://localhost:8080/rest/private/cmisatom/Managed%20Sites/typebyid/exo%3Ataxonomy" rel="describ
   <link href="http://localhost:8080/rest/private/cmisatom/Managed%20Sites/allowableactions/f59a3539c0a800036257
   <link href="http://localhost:8080/rest/private/cmisatom/Managed%20Sites/relationships/f59a3539c0a80003625790b
   <link href="http://localhost:8080/rest/private/cmisatom/Managed%20Sites/policies/f59a3539c0a80003625790bdadf5
   <link href="http://localhost:8080/rest/private/cmisatom/Managed%20Sites/objacl/f59a3539c0a80003625790bdadf566
   <link href="http://localhost:8080/rest/private/cmisatom/Managed%20Sites/children/f59a3539c0a80003625790bdadf5
   <link href="http://localhost:8080/rest/private/cmisatom/Managed%20Sites/descendants/f59a3539c0a80003625790bda
   <link href="http://localhost:8080/rest/private/cmisatom/Managed%20Sites/foldertree/f59a3539c0a80003625790bdac
   <link href="http://localhost:8080/rest/private/cmisatom/Managed%20Sites/object/f59a3533c0a8000339af97059f243a
   <content type="text">News</content>
   <cmisra:object xmlns:cmis="http://docs.oasis-open.org/ns/cmis/core/200908/">
      <cmis:properties>
         <cmis:propertyId displayName="cmis:allowedChildObjectTypeIds" localName="cmis:allowedChildObjectTypeIds
         <cmis:propertyString displayName="cmis:path" localName="cmis:path" propertyDefinitionId="cmis:path" que
            <cmis:value>/acme/categories/acme/News</cmis:value>
         </cmis:propertyString>
         <cmis:propertyId displayName="cmis:objectTypeId" localName="cmis:objectTypeId" propertyDefinitionId="cm
            <cmis:value>exo:taxonomy</cmis:value>
         </cmis:propertyId>
         <cmis:propertyString displayName="cmis:lastModifiedBy" localName="cmis:lastModifiedBy" propertyDefiniti
         <cmis:propertyString displayName="cmis:name" localName="cmis:name" propertyDefinitionId="cmis:name" que
            <cmis:value>News</cmis:value>
         </cmis:propertyString>
         <cmis:propertyString displayName="cmis:createdBy" localName="cmis:createdBy" propertyDefinitionId="cmis
         <cmis:propertyId displayName="cmis:objectId" localName="cmis:objectId" propertyDefinitionId="cmis:objec
            <cmis:value>f59a3539c0a80003625790bdadf566c5</cmis:value>
         </cmis:propertyId>
         <cmis:propertyDateTime displayName="cmis:creationDate" localName="cmis:creationDate" propertyDefinitior
         <cmis:propertyString displayName="cmis:changeToken" localName="cmis:changeToken" propertyDefinitionId="
         <cmis:propertyId displayName="cmis:baseTypeId" localName="cmis:baseTypeId" propertyDefinitionId="cmis:b
            <cmis:value>cmis:folder</cmis:value>
         </cmis:propertyId>
         <cmis:propertyId displayName="cmis:parentId" localName="cmis:parentId" propertyDefinitionId="cmis:parer
            <cmis:value>f59a3533c0a8000339af97059f243a25</cmis:value>
         </cmis:propertyId>
         <cmis:propertyDateTime displayName="cmis:lastModificationDate" localName="cmis:lastModificationDate" pr
      </cmis:properties>
      <cmis:acl/>
      <cmis:exactACL>false</cmis:exactACL>
```

```
      <cmis:policyIds/>
      <cmis:rendition/>
   </cmisra:object>
</entry>
```

To get the file which has uploaded above, use the **children** service to get the list of child nodes of /acme/documents. Find this service URL in the response XML above:

```
curl -o childs.xml -u root:gtn http://localhost:8080/rest/private/cmisatom/Managed%20Sites/children/f59a3539c0a8
```

In the requested file (childs.xml), find the entry related to **test.txt** file uploaded via **Sites Explorer**. Find by title for name "test.txt".

```
   <entry xmlns:cmisra="http://docs.oasis-open.org/ns/cmis/restatom/200908/">
      <id>f708e208c0a80003554babb97bd934ba</id>
      <published>2010-09-09T18:06:31.987Z</published>
      <updated>2010-09-09T18:06:31.987Z</updated>
      <summary type="text"/>
      <author>
         <name>system</name>
      </author>
      <title type="text">test.txt</title>
      <link href="http://localhost:8080/rest/private/cmisatom/Managed%20Sites" rel="service" type="application/a
      <link href="http://localhost:8080/rest/private/cmisatom/Managed%20Sites/object/f708e208c0a80003554babb97bd
      <link href="http://localhost:8080/rest/private/cmisatom/Managed%20Sites/object/f708e208c0a80003554babb97bd
      <link href="http://localhost:8080/rest/private/cmisatom/Managed%20Sites/typebyid/cmis%3Adocument" rel="des
      <link href="http://localhost:8080/rest/private/cmisatom/Managed%20Sites/allowableactions/f708e208c0a800035
      <link href="http://localhost:8080/rest/private/cmisatom/Managed%20Sites/relationships/f708e208c0a80003554b
      <link href="http://localhost:8080/rest/private/cmisatom/Managed%20Sites/policies/f708e208c0a80003554babb97
      <link href="http://localhost:8080/rest/private/cmisatom/Managed%20Sites/objacl/f708e208c0a80003554babb97bd
      <link href="http://localhost:8080/rest/private/cmisatom/Managed%20Sites/versions/f708a0f1c0a8000333e3681f9
      <link href="http://localhost:8080/rest/private/cmisatom/Managed%20Sites/object/f708e208c0a80003554babb97bd
      <link href="http://localhost:8080/rest/private/cmisatom/Managed%20Sites/parents/f708e208c0a80003554babb97b
      <link href="http://localhost:8080/rest/private/cmisatom/Managed%20Sites/file/f708e208c0a80003554babb97bd93
      <content src="http://localhost:8080/rest/private/cmisatom/Managed%20Sites/file/f708e208c0a80003554babb97bd
      <cmisra:object xmlns:cmis="http://docs.oasis-open.org/ns/cmis/core/200908/">
         <cmis:properties>
            <cmis:propertyBoolean displayName="cmis:isLatestMajorVersion" localName="cmis:isLatestMajorVersion"
               <cmis:value>false</cmis:value>
            </cmis:propertyBoolean>
            <cmis:propertyInteger displayName="cmis:contentStreamLength" localName="cmis:contentStreamLength" pr
               <cmis:value>38</cmis:value>
            </cmis:propertyInteger>
            <cmis:propertyId displayName="cmis:contentStreamId" localName="cmis:contentStreamId" propertyDefinit
               <cmis:value>f708a0c2c0a800033bedeb35caddeed1</cmis:value>
            </cmis:propertyId>
            <cmis:propertyId displayName="cmis:objectTypeId" localName="cmis:objectTypeId" propertyDefinitionId=
               <cmis:value>cmis:document</cmis:value>
            </cmis:propertyId>
            <cmis:propertyString displayName="cmis:versionSeriesCheckedOutBy" localName="cmis:versionSeriesCheck
            <cmis:propertyId displayName="cmis:versionSeriesCheckedOutId" localName="cmis:versionSeriesCheckedOu
            <cmis:propertyString displayName="cmis:name" localName="cmis:name" propertyDefinitionId="cmis:name"
               <cmis:value>test.txt</cmis:value>
            </cmis:propertyString>
            <cmis:propertyString displayName="cmis:contentStreamMimeType" localName="cmis:contentStreamMimeType"
               <cmis:value>text/plain</cmis:value>
            </cmis:propertyString>
            <cmis:propertyId displayName="cmis:versionSeriesId" localName="cmis:versionSeriesId" propertyDefinit
               <cmis:value>f708a0f1c0a8000333e3681f99fab760</cmis:value>
            </cmis:propertyId>
            <cmis:propertyDateTime displayName="cmis:creationDate" localName="cmis:creationDate" propertyDefinit
               <cmis:value>2010-09-09T18:06:31.987Z</cmis:value>
            </cmis:propertyDateTime>
            <cmis:propertyString displayName="cmis:changeToken" localName="cmis:changeToken" propertyDefinitionI
            <cmis:propertyBoolean displayName="cmis:isLatestVersion" localName="cmis:isLatestVersion" propertyDe
               <cmis:value>true</cmis:value>
            </cmis:propertyBoolean>
            <cmis:propertyString displayName="cmis:versionLabel" localName="cmis:versionLabel" propertyDefinitio
               <cmis:value>latest</cmis:value>
            </cmis:propertyString>
            <cmis:propertyBoolean displayName="cmis:isVersionSeriesCheckedOut" localName="cmis:isVersionSeriesCh
               <cmis:value>false</cmis:value>
```

```
            </cmis:propertyBoolean>
            <cmis:propertyString displayName="cmis:lastModifiedBy" localName="cmis:lastModifiedBy" propertyDefin
            <cmis:propertyString displayName="cmis:createdBy" localName="cmis:createdBy" propertyDefinitionId="c
            <cmis:propertyString displayName="cmis:checkinComment" localName="cmis:checkinComment" propertyDefin
            <cmis:propertyId displayName="cmis:objectId" localName="cmis:objectId" propertyDefinitionId="cmis:ob
               <cmis:value>f708e208c0a80003554babb97bd934ba</cmis:value>
            </cmis:propertyId>
            <cmis:propertyBoolean displayName="cmis:isImmutable" localName="cmis:isImmutable" propertyDefinitior
               <cmis:value>false</cmis:value>
            </cmis:propertyBoolean>
            <cmis:propertyBoolean displayName="cmis:isMajorVersion" localName="cmis:isMajorVersion" propertyDefi
               <cmis:value>false</cmis:value>
            </cmis:propertyBoolean>
            <cmis:propertyId displayName="cmis:baseTypeId" localName="cmis:baseTypeId" propertyDefinitionId="cmi
               <cmis:value>cmis:document</cmis:value>
            </cmis:propertyId>
            <cmis:propertyString displayName="cmis:contentStreamFileName" localName="cmis:contentStreamFileName"
               <cmis:value>test.txt</cmis:value>
            </cmis:propertyString>
            <cmis:propertyDateTime displayName="cmis:lastModificationDate" localName="cmis:lastModificationDate"
               <cmis:value>2010-09-09T18:06:31.987Z</cmis:value>
            </cmis:propertyDateTime>
         </cmis:properties>
         <cmis:acl/>
         <cmis:exactACL>false</cmis:exactACL>
         <cmis:policyIds/>
         <cmis:rendition/>
      </cmisra:object>
   </entry>
```

Then, get the **test.txt** file content via CMIS by using the **file** service and **id** of the file "test.txt" from **childs.xml**:

```
curl -o test.txt -u root:gtn http://localhost:8080/rest/private/cmisatom/Managed%20Sites/file/f708e208c0a8000355
```

Get results in the test.txt file in the local folder. In this way we got file stored in **Sites Explorer** to folder /acme/documents/test.txt via **eXo CMIS** by symlink path /acme/categories/acme/News/test.txt. As it seen file's actual content referenced by id in CMIS. Path has no matter for content read or change.

4.3.4. Modify WCM via CMIS

Note

Upload the modified local file using the command with id of the file stored in WCM (it's jcr:uuid of the file in JCR Workspace). Note that you should run this command from the folder where the local file is stored.

To update any file via CMIS, use the **file** service and the PUT request. Use the same file as in the previous usecase (/acme/documents/test.txt, id:f708e208c0a80003554babb97bd934ba).

```
curl -T test.txt -X PUT -H "Content-Type:text/plain; charset=UTF-8" -u root:gtn http://localhost:8080/rest/priva
```

Go to the **Sites Explorer** and see changes applied from the local file in /acme/documents/test.txt.

4.3.5. CMIS search

CMIS provides a type-based query service for discovering objects that match specified criteria, by defining a read-only projection of the CMIS data model into a Relational View.

CMIS query languages is based on a subset of the SQL-92 grammar. CMIS-specific language extensions to SQL-92 are called out explicitly. The basic structure of a CMIS query is a SQL statement that MUST include the following clauses:

- SELECT (virtual columns): This clause identifies the set of virtual columns that will be included in the query results for each row.
- FROM (Virtual Table Names): This clause identifies which Virtual Table(s) the query will run against.

Additionally, a CMIS query MAY include the following clauses:

- WHERE (conditions): This clause identifies the constraints that rows MUST satisfy to be considered a result for the query.
- ORDER BY (sort specification): This clause identifies the order in which the result rows MUST be sorted in the result row set.

Each CMIS ObjectType definition has next query attributes:

- query name (String) - Used for query operations on object types. In our SQL statement examples all objecttypes is a queryName, i.e. the given queryName matches the specific type of document. For example, in query like "SELECT * FROM cmis:document" ,"cmis:document" is queryName preconfigured in Document object type definition.
- queryable (Boolean) - Indicates whether or not this object type is queryable. A non-queryable object type is not visible through the relational view that is used for query, and can not appear in the FROM clause of a query statement.
- fulltextIndexed (Boolean) - Indicates whether objects of this type are full-text indexed for querying via the CONTAINS() query predicate.
- includedInSupertypeQuery (Boolean) - Indicates whether this type and its subtypes appear in a query of this type's ancestor types. For example: if Invoice is a sub-type of Document, if this is TRUE on Invoice then for a query on Document type, instances of Invoice will be returned if they match. If this attribute is FALSE, no instances of Invoice will be returned even if they match the query.

Property definition also contains queryName and queriable attributes with same usage.

4.3.5.1. Query examples

4.3.5.1.1. Simple query

Query : Select all cmis:document.

```
SELECT * FROM cmis:document
```

Query result:

- All documents from Apollo program.

4.3.5.1.2. Find document by several constraints

Query : Select all documents where apollo:propertyBooster is 'Saturn V' and apollo:propertyCommander is Frank F. Borman, II or James A. Lovell, Jr.

Initial data:

- document1: apollo:propertyBooster - Saturn 1B, apollo:propertyCommander - Walter M. Schirra
- document2: apollo:propertyBooster - Saturn V, apollo:propertyCommander - Frank F. Borman, II
- document3: apollo:propertyBooster - Saturn V, apollo:propertyCommander - James A. Lovell, Jr.

```
SELECT * FROM cmis:document WHERE apollo:propertyBooster = 'Saturn V' AND (apollo:propertyCommander = 'Frank F.
```

Query result:

- document2 and document3.

4.3.5.1.3. Full-text search

Query : Select all documents that contains "here" word.

Initial data:

- document1: content - "There must be test word"
- document2: content - "Test word is not here"

```
SELECT * FROM cmis:document WHERE CONTAINS('here')
```

Query result:

- document2.

4.3.5.1.4. Extended full-text search

Query : Select all documents that contains "There must" phrase and do not contain "check-word" word.

Initial data:

- document1: content - "There must be test word."
- document2: content - "Test word is not here. Another check-word."
- document3: content - "There must be check-word."

```
SELECT * FROM cmis:document WHERE CONTAINS("\"There must\" -\"check\-word\"")
```

Query result:

- document1.

4.3.5.1.5. Date property comparison

Query : Select all documents where cmis:lastModificationDate more than 2007-01-01.

Initial data:

- document1: cmis:lastModificationDate - 2006-08-08
- document2: cmis:lastModificationDate - 2009-08-08

```
SELECT * FROM cmis:document WHERE (cmis:lastModificationDate >= TIMESTAMP '2007-01-01T00:00:00.000Z')"
```

Query result:

- document2.

4.3.5.1.6. Boolean property comparison

Query : Select all documents where property apollo:someProperty equals to false.

Initial data:

- document1: apollo:someProperty - true
- document2: apollo:someProperty - false

```
SELECT * FROM cmis:document WHERE (apollo:someProperty = FALSE)
```

Query result:

- document2.

4.3.5.1.7. IN Constraint

Query : Select all documents where apollo:propertyCommander is in set {'Virgil I. Grissom', 'Frank F. Borman, II', 'James A. Lovell, Jr.'}.

Initial data:

- document1: apollo:propertyCommander - Walter M. Schirra
- document2: apollo:propertyCommander - Frank F. Borman, II
- document3: apollo:propertyCommander - James A. Lovell, Jr.
- document4: apollo:propertyCommander - Eugene A. Cernan

```
SELECT * FROM cmis:document WHERE apollo:propertyCommander IN ('Virgil I. Grissom', 'Frank F. Borman, II', 'Jame
```

Query result:

- document2 and document3.

4.3.5.1.8. Select all documents where longprop property NOT IN set

Query : Select all documents where apollo:propertyCommander property not in set {'Walter M. Schirra', 'James A. Lovell, Jr.'}.

Initial data:

- document1: apollo:propertyCommander - Walter M. Schirra
- document2: apollo:propertyCommander - Frank F. Borman, II
- document3: apollo:propertyCommander - James A. Lovell, Jr.
- document4: apollo:propertyCommander - Eugene A. Cerna

```
SELECT * FROM cmis:document WHERE apollo:PropertyCommander NOT IN ('Walter M. Schirra', 'James A. Lovell, Jr.')
```

Query result:

- document2,document4.

4.3.5.1.9. Select all documents where longprop property NOT NOT IN set

Query : Select all documents where apollo:propertyCommander property NOT IN set {'James A. Lovell, Jr.'}.

Initial data:

- document1: apollo:propertyCommander - Walter M. Schirra
- document2: apollo:propertyCommander - Frank F. Borman, II
- document3: apollo:propertyCommander - James A. Lovell, Jr.
- document4: apollo:propertyCommander - Eugene A. Cerna

```
SELECT * FROM cmis:document WHERE  NOT (apollo:propertyCommander NOT IN ('James A. Lovell, Jr.'))
```

Query result:

- document3.

4.3.5.1.10. IN_FOLDER constarint

Query : Select all folders that are in folder1.

Initial data:

- folder1: id - 123456789
  - document1: Title - node1
- folder3:
  - folder4:
- folder2:
  - document2: Title - node2

```
SELECT * FROM cmis:folder WHERE IN_FOLDER('123456789')
```

Query result:

- folder3.

4.3.5.1.11. Select all documents that are in specified folder

Query : Select all documents that are in folder1.

Initial data:

- folder1: id - 123456789
  - document1: Title - node1
- folder2:
  - document2: Title - node2

```
SELECT * FROM cmis:document WHERE IN_FOLDER('123456789')
```

Query result:

- document1.

4.3.5.1.12. Select all documents where query supertype is cmis:article

Initial data:

- testRoot: id - 123456789
- document1: Title - node1 typeID - cmis:article-sports
- document2: Title - node2 typeID - cmis:article-animals

```
SELECT * FROM cmis:article WHERE IN_FOLDER('123456789')
```

Query result:

- document1,document2.

4.3.5.1.13. IN_TREE constraint

Query : Select all documents that are in tree of folder1.

Initial data:

- folder1: id - 123456789
  - document1
- folder2:
  - document2

```
SELECT * FROM cmis:document WHERE IN_TREE('123456789')
```

Query result:

- document1,document2.

4.3.5.1.14. LIKE Comparison

Query : Select all documents where apollo:propertyCommander begins with "James".

Initial data:

- document1: apollo:propertyCommander - Walter M. Schirra
- document2: apollo:propertyCommander - Frank F. James, II
- document3: apollo:propertyCommander - James A. Lovell, Jr.
- document4: apollo:propertyCommander - Eugene A. James

```
SELECT * FROM cmis:document AS doc WHERE apollo:PropertyCommander LIKE 'James%'
```

Query result:

- document3.

4.3.5.1.15. Test LIKE constraint with escape symbols

Query : Select all documents where apollo:someProperty like 'admin%'.

Initial data:

- document1: Title - node1, apollo:someProperty - ad%min master
- document2: Title - node2, apollo:someProperty - admin operator
- document3: Title - node2, apollo:someProperty - radmin

```
SELECT * FROM cmis:document AS doc WHERE apollo:someProperty  LIKE 'ad\%min%'
```

Query result:

- document1.

4.3.5.1.16. NOT constraint

Query : Select all documents that not contains "world" word.

Initial data:

- document1: Title - node1, content - hello world
- document2: Title - node2, content - hello

```
SELECT * FROM cmis:document WHERE NOT CONTAINS('world')
```

Query result:

- document2.

4.3.5.1.17. Property existence

Query : Select all documents that has apollo:propertyCommander property IS NOT NULL.

Initial data:

- document1: apollo:propertyCommander - Walter M. Schirra
- document2: apollo:propertyCommander -
- document3: apollo:propertyCommander - James A. Lovell, Jr.
- document4: apollo:propertyCommander -

```
SELECT * FROM cmis:document WHERE apollo:propertyCommander IS NOT NULL
```

Query result:

- document1,document3.

4.3.5.1.18. ORDER BY

Query : Select all documents in default order (by document name).

Initial data:

- document1: Title - Apollo 7
- document2: Title - Apollo 8
- document3: Title - Apollo 13
- document4: Title - Apollo 17

```
SELECT cmis:lastModifiedBy, cmis:objectId, cmis:lastModificationDate FROM cmis:document
```

Query result:

- document3,document4,document1,document2.

4.3.5.1.19. ORDER BY ASC

Query : Order by apollo:propertyCommander property value (in ascending order).

Initial data:

- document1: apollo:propertyCommander - Walter M. Schirra
- document2: apollo:propertyCommander - Frank F. Borman, II
- document3: apollo:propertyCommander - James A. Lovell, Jr.
- document4: apollo:propertyCommander - Eugene A. Cerna

```
SELECT cmis:lastModifiedBy, cmis:objectId, cmis:lastModificationDate FROM cmis:document ORDER BY apollo:property
```

Query result:

- document4,document2,document3,document1.

4.3.5.1.20. ORDER BY DESC

Query : Order by apollo:propertyCommander property value (in decending order).

Initial data:

- document1: apollo:propertyCommander - Walter M. Schirra
- document2: apollo:propertyCommander - Frank F. James, II
- document3: apollo:propertyCommander - James A. Lovell, Jr.
- document4: apollo:propertyCommander - Eugene A. James

```
SELECT cmis:lastModifiedBy, cmis:objectId, cmis:lastModificationDate FROM cmis:document ORDER BY cmis:propertyCo
```

Query result:

- document1,document3,document2,document4.

4.3.5.1.21. ORDER BY SCORE (as columns)

Query : Select all documents which contains word "moon" ordered by score.

Initial data:

- document1: content - "Earth-orbital mission, the first manned launch"
- document2: content - "from another celestial body - Earth's Moon"
- document3: content - "NASA intended to land on the Moon, but a mid-mission technical"
- document4: content - "It was the first night launch of a U.S. human"

```
SELECT cmis:lastModifiedBy, cmis:objectId, cmis:lastModificationDate FROM cmis:document WHERE CONTAINS('moon') O
```

Query result:

- document2,document3.

4.3.5.1.22. Not equal comparison (decimal)

Query : Select all documents property apollo:propertyBooster not equal to 3.

Initial data:

- document1: Title - node1, apollo:propertyBooster - 3
- document2: Title - node2, apollo:propertyBooster - 15

```
SELECT * FROM cmis:document WHERE apollo:propertyBooster <> 3
```

Query result:

- document2.

4.3.5.1.23. Not equal comparison (string)

Query : Select all documents property apollo:someProperty not equals to "test word second".

Initial data:

- document1: apollo:someProperty - "test word first"

- document2: apollo:someProperty - "test word second"

```
SELECT * FROM cmis:document WHERE apollo:someProperty <> 'test word second'
```

Query result:

- document1.

4.3.5.1.24. More than comparison (>)

Query : Select all documents property apollo:propertyBooster more than 5.

Initial data:

- document1: apollo:propertyBooster - 3
- document2: apollo:propertyBooster - 15

```
SELECT * FROM cmis:document WHERE apollo:propertyBooster > 5
```

Query result:

- document2.

Chapter 5. CMIS Usage code examples

The following examples of CMIS usage may be useful for developers who needs to access a repository. CMIS access code snippets build using Apache HTTP Client for Java or using Google gadgets (gadgets.io) for JavaScript examples. For the cURL examples look at http://code.google.com/p/xcmis/wiki/xCMISusesWithCurl.

5.1.1. Login to repository

Note

CMIS service uses default authentication in general case, but it can be overridden in case of embedding CMIS into an Application Service. In these examples only Basic HTTP authentication covered.

5.1.1.1. Using java

```
import org.apache.commons.httpclient.HttpClient;
import org.apache.commons.httpclient.UsernamePasswordCredentials;
import org.apache.commons.httpclient.auth.AuthScope;
import org.apache.commons.httpclient.methods.GetMethod;

HttpClient client = new HttpClient();
client.getState().setCredentials(
    new AuthScope("localhost", 8080, "realm"),
    new UsernamePasswordCredentials("root", "exo");
....
```

5.1.2. Listing of documents (folder, files)

There are a several methods to get the documents listings, such as getChildren(), getFolderTree() and getDescentants(), their usage will be described below. The difference between them is in usage of different URL segments to get a data ("/children" for getChildren(), "/foldertree" for getFolderTree(), "/descendants" for getDescentants()), and a different kind of results (getChildren() returns a flat structure, while a getFolderTree() and getDescentants() has a tree of items in response).

5.1.2.1. Using java

```
import org.apache.commons.httpclient.HttpClient;
import org.apache.commons.httpclient.methods.GetMethod;
import org.apache.commons.httpclient.MultiThreadedHttpConnectionManager;

String url = "http://localhost:8080/rest/private/cmisatom/";
url += repository;
url += "/children/";
url += obj_id;

HttpClient client = new HttpClient(new MultiThreadedHttpConnectionManager());
client.getHttpConnectionManager().
getParams().setConnectionTimeout(10000);

GetMethod get = new GetMethod(url);
try {
int result = client.executeMethod(get);
final String strResponse = get.getResponseBodyAsString();
} finally {
  get.releaseConnection();
```

```
}
```

5.1.2.2. Using JavaScript

Creating an URL to make a request (consists of repository name, the method name, e.g. "/children/", and folderID to get children from):

```
var url = "http://localhost:8080/rest/private/cmisatom/";
url += repository;
url += "/children/";
url += obj_id;
```

performing request:

```
var params = {};
params[gadgets.io.RequestParameters.METHOD] = gadgets.io.MethodType.GET;
params[gadgets.io.RequestParameters.CONTENT_TYPE] = gadgets.io.ContentType.FEED;
gadgets.io.makeRequest(url, handler, params);
```

processing results (code located in the handler specified while making a request, the same way it might be used for all examples in this chapter):

```
var handler = function(resp) {
  var data = eval(resp.data.Entry);
   for (var i = 0; i < data.length; i++) {
     var doc = data[i];
     alert(doc.Title);
     alert(doc.Date);
      ...etc..
   }
}
```

5.1.3. Read document properties and content-stream

Reading of Document properties and content stream it are two separate operations. Getting of content stream is possible after the properties set have been read and the content stream ID extracted from it.

5.1.3.1. Using java

Get document properties.

```
import org.apache.commons.httpclient.HttpClient;
import org.apache.commons.httpclient.methods.GetMethod;
import org.apache.commons.httpclient.MultiThreadedHttpConnectionManager;

String url = "http://localhost:8080/rest/private/cmisatom/";
url += repository;
url += "/object/";
url += obj_id;

HttpClient client = new HttpClient(new MultiThreadedHttpConnectionManager());
client.getHttpConnectionManager().
getParams().setConnectionTimeout(10000);

GetMethod get = new GetMethod(url);
try {
  int result = client.executeMethod(get);
  final String strResponse = get.getResponseBodyAsString();
  // use response...
```

```
} finally {
  get.releaseConnection();
}
```

Get document content-stream.

To get a Document's content stream, an URL must contain "/file" part, object ID, and optionally the content stream ID, which can be used, for example, to obtain renditions. If no stream ID is specified, a default stream will be returned.

```
import org.apache.commons.httpclient.HttpClient;
import org.apache.commons.httpclient.methods.PostMethod;

String url = "http://localhost:8080/rest/private/cmisatom/";
url += repository;
url += "/file/";
url += obj_id;
//Optionally
url += "?";
url += "streamid=";
url += streamID;

HttpClient client = new HttpClient();
client.getHttpConnectionManager().
getParams().setConnectionTimeout(10000);

GetMethod get = new GetMethod(url);
try {
  int result = client.executeMethod(get);
  final InputStream stream = get.getResponseBodyAsStream();
  try {
  // use stream...
  int dataByte = stream.read();
  } finally {
    stream.close();
  }
} finally {
  get.releaseConnection();
}
```

5.1.3.2. Using JavaScript.

Get document properties.

Creating an URL to make a request (consists of repository name, method name, e.g. "/children/", and folder ID to get the children from):

```
var url = "http://localhost:8080/rest/private/cmisatom/";
url += repository;
url += "/object/";
url += obj_id;
```

performing request:

```
var params = {};
params[gadgets.io.RequestParameters.METHOD] = gadgets.io.MethodType.GET;
params[gadgets.io.RequestParameters.CONTENT_TYPE] = gadgets.io.ContentType.FEED;
gadgets.io.makeRequest(url, handler, params);
```

You can also use the ContentType.DOM parameter to parse the feed in your application (Using DOMParser for example).

Get document content-stream.

Note

Performing a content stream request in JavaScript will cause the browser dialog for a file download.

```
var url = "http://localhost:8080/rest/private/cmisatom/";
url += repository;
url += "/file/";
url += obj_id;
//Optionally
url += "?";
url += "streamid=";
url += streamID;
```

5.1.4. Search of data and syntax examples

CMIS supports SQL queries for more handful content search. Query service can handle both GET and POST requests. URL for query consists of repository name and method name "/query". GET request must contain query as a parameter named "q", in case of POST request query must be located in a request body.

For more detailed instructions how to construct queries please refer to "Query examples" chapter.

5.1.4.1. Using java

```
import org.apache.commons.httpclient.HttpClient;
import org.apache.commons.httpclient.methods.PostMethod;
import org.apache.commons.httpclient.methods.StringRequestEntity;

String url = "http://localhost:8080/rest/private/cmisatom/";
url += repository;
url += "/query/";

HttpClient client = new HttpClient();
client.getHttpConnectionManager().
getParams().setConnectionTimeout(10000);

PostMethod post = new PostMethod(url);
String s = "<?xml version='1.0' encoding='utf-8'?>"
        + "<cmis:query xmlns='http://www.w3.org/2005/Atom' xmlns:cmis='http://docs.oasis-open.org/ns/cmis/core/
        + "<cmis:statement>SELECT * FROM cmis:document</cmis:statement>"
        + "<cmis:maxItems>10</cmis:maxItems>"
        + "<cmis:skipCount>0</cmis:skipCount>"
        + "<cmis:searchAllVersions>true</cmis:searchAllVersions>"
        + "<cmis:includeAllowableActions>true</cmis:includeAllowableActions>"
        + "</cmis:query>";

RequestEntity entity = new StringRequestEntity(s, "text/xml","utf-8");
try {
  post.setRequestEntity(entity);
  int result = client.executeMethod(post);
  final String strResponse = post.getResponseBodyAsString();
  // use response...
} finally {
  post.releaseConnection();
}
```

5.1.4.2. Using JavaScript

```
var url = "http://localhost:8080/rest/private/cmisatom/";
```

```
url += repository;
url += "/query/";
```

```
var params = {};
params[gadgets.io.RequestParameters.METHOD] = gadgets.io.MethodType.POST;
params[gadgets.io.RequestParameters.POST_DATA] = gadgets.io.encodeValues(someQuery);
gadgets.io.makeRequest(url, handler, params);
```

5.1.5. Modification of document properties or content

Command of property update uses PUT method. The URL is the same as for a reading of properties, the difference is only in HTTP method used. Body of the request must be an Atom document with the properties specified (see spec. 2.2.4.12 for details constructing document).

Sending of content stream can be executed via PUT or POST requests. Content-type of the request must be an "multipart/form-data".

5.1.5.1. Using java

Update properties:

```
import org.apache.commons.httpclient.HttpClient;
import org.apache.commons.httpclient.methods.StringRequestEntity;
import org.apache.commons.httpclient.methods.PostMethod;
import org.apache.commons.httpclient.methods.RequestEntity;

String url = "http://localhost:8080/rest/private/cmisatom/";
url += repository;
url += "/object/";
url += obj_id;

HttpClient client = new HttpClient();
client.getHttpConnectionManager().
getParams().setConnectionTimeout(10000);

String atomDoc = "<?xml version='1.0' encoding='utf-8'?>"
        + "<entry xmlns='http://www.w3.org/2005/Atom'"
        + " xmlns:cmis='http://docs.oasis-open.org/ns/cmis/core/200908/'"
        + " xmlns:cmisra='http://docs.oasis-open.org/ns/cmis/restatom/200908/'>"
        + "<cmisra:object><cmis:properties>"
        + "<cmis:propertyString queryName='cmis:name' localName='cmis:name' propertyDefinitionId='cmis:name'>"
        + "<cmis:value>newName</cmis:value>"
        + "</cmis:propertyString>"
        + "</cmis:properties></cmisra:object>"
        + "</entry>";

PutMethod put = new PutMethod(url);
RequestEntity entity = new StringRequestEntity(atomDoc, "text/xml", "utf-8");
put.setRequestEntity(entity);

try {
  int result = client.executeMethod(put);
  final String strResponse = put.getResponseBodyAsString();
} finally {
  put.releaseConnection();
}
```

Set content stream:

```
import org.apache.commons.httpclient.HttpClient;
import org.apache.commons.httpclient.methods.InputStreamRequestEntity;
import org.apache.commons.httpclient.methods.PostMethod;
import org.apache.commons.httpclient.methods.RequestEntity;
```

```
String url = "http://localhost:8080/rest/private/cmisatom/";
url += repository;
url += "/file/";
url += obj_id;

HttpClient client = new HttpClient();
client.getHttpConnectionManager().
getParams().setConnectionTimeout(10000);

PostMethod post = new PostMethod(url);
RequestEntity entity = new InputStreamRequestEntity(inputStream, "text/xml; charset=ISO-8859-1");
post.setRequestEntity(entity);

try {
  int result = client.executeMethod(post);
  final String strResponse = post.getResponseBodyAsString();
} finally {
  post.releaseConnection();
}
```

5.1.5.2. Using JavaScript

Update properties:

```
var url = "http://localhost:8080/rest/private/cmisatom/";
url += repository;
url += "/object/";
url += obj_id;
```

```
//constructing document
 String atomDoc = "<?xml version='1.0' encoding='utf-8'?>";
       atomDoc += "<entry xmlns='http://www.w3.org/2005/Atom'";
       atomDoc += " xmlns:cmis='http://docs.oasis-open.org/ns/cmis/core/200908/'";
       atomDoc += " xmlns:cmisra='http://docs.oasis-open.org/ns/cmis/restatom/200908/'>";
       atomDoc += "<cmisra:object><cmis:properties>";
       atomDoc += "<cmis:propertyString queryName='cmis:name' localName='cmis:name' propertyDefinitionId='cmis:r
       atomDoc += "<cmis:value>newName</cmis:value>";
       atomDoc += "</cmis:propertyString>";
       atomDoc += "</cmis:properties></cmisra:object></entry>";

 var params = {};
 params[gadgets.io.RequestParameters.METHOD] = gadgets.io.MethodType.PUT;
 params[gadgets.io.RequestParameters.POST_DATA] = atomDoc;
 gadgets.io.makeRequest(url, handler, params);
```

Set content stream:

```
var url = "http://localhost:8080/rest/private/cmisatom/";
url += repository;
url += "/file/";
url += obj_id;
```

```
var params = {};
params[gadgets.io.RequestParameters.METHOD] = gadgets.io.MethodType.POST;
params[gadgets.io.RequestParameters.CONTENT_TYPE] = "multipart/form-data";
params[gadgets.io.RequestParameters.POST_DATA] = contentStream;
gadgets.io.makeRequest(url, handler, params);
```

Chapter 6. References

xCMIS project links:

- Community site http://xcmis.org/.
- JIRA site https://jira.exoplatform.org/browse/CMIS.
- Forum http://groups.google.com/group/xcmis.
- xCMIS latest news http://gazarenkov.blogspot.com/.
- eXo Platform blog http://blog.exoplatform.org/.
- http://xcmis.org/CmisExpert1/org.exoplatform.cmis.CmisExpertApplication/CmisExpertApplication.html .

CMIS-related links:

- http://www.oasis-open.org/committees/tc_home.php?wg_abbrev=cmis .
- TODO CMIS working group (list, mail, feeds etc).
- TODO CMIS external tutorials, guides, videos, blogs, news.
- TODO CMIS clients, UI, web.